Introdução ao LaTeX

Seminário LaTeX - o Livro

Geraldo Xexéo[1,2]

[1]Departamento de Ciências da Computação
[2]Programa de Engenharia de Sistemas e Computação

Março 2020

Sumário

Resumo

Esse texto é uma introdução ao LaTeXescrita em português, criada como resultado de um seminário de introdução ao LaTeXrealizado na época do Covid-19.

Ele não pretende ser a melhor introdução ao LaTeX, apenas mais uma, mas foi criada com a intenção de facilitar um pouco a vida dos meus alunos.

Capítulo 1

Introdução ao Processamento de Texto

Os computadores foram criados para fazer contas, mas, na verdade, eles manipulam apenas símbolos. Rapidamente seus usuários perceberam que podiam ser usados para manipular textos, como código, e formatá-los de alguma forma para impressão. O processamento de texto é possivelmente a área de software com o maior número de usuários, já que todo usuário de computador, alguma vez, escreveu algo e enviou para alguém, nem que seja um simples e-mail. Para isso, foram criados vários tipos de programas, que cumprem funções como as descritas na figura 1.1, criando o que pode ser chamado de uma cadeia de processamento de texto. Basicamente, o que a figura mostra é um arquivo de texto pode ser editado, visualizado, processado para impressão ou processado de formas adicionais. Cada sistema de processamento de texto faz, prioritariamente uma dessas funções[1].

1.1 Tipos de Sistema de Processamento de Texto

Dentro dessa cadeia de processamento existem vários tipos de programas. Os principais tipos são:

- Editores de Texto
- IDEs
- Processadores de Texto

[1]Observa-se que esta cadeia pode, para outros contextos, ser representada de forma mais complexa, incluindo conceito de produção, versionamento, seguração, etc.

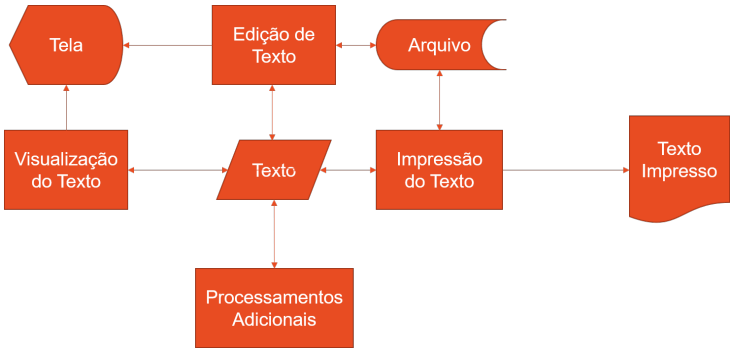

Figura 1.1: A cadeia de processamento de texto

- Sistemas de Autoria
- Sistemas de Publicação (*Desktop Publishing*)
- Sistemas de Composição
- Sistemas Colaborativos de Edição

1.1.1 Editores de Texto

Editores de texto são programas que permitem ao usuário editar arquivos que são de **texto puro**. Texto puro é um conceito que mudou. Inicialmente significava que havia um mapeamento um para um entre o que você encontrava no arquivo, uma sequência de caracteres codificados em ASCII[2]. Em ASCII, por exemplo, a letra "A" é representada pelos bits "1000001" e a "a" por "1100001". Atualmente são usadas codificações que permitem que um caracter ou símbolo seja representado por uma sequência mais longa de bits, por meio de *escape codes*, por exemplo UTF-8[3], o que significa que os editores de texto, normalmente, não representam mais perfeitamente o arquivo em disco. Os sistemas de codificação atuais são normalmente extensões do ASCII, isto é, os 128 códigos de 1 byte do ASCII original ainda são válidos. A figura 1.2 mostra a função de um editor de texto na cadeia de processamento de texto.

Editores de texto foram necessários assim que trocamos as entradas por cartão e fita, que eram editados em máquinas não conectadas ao computador,

[2]ASCII é um padrão que associa uma letra, e outros símbolos usados em arquivos, a um byte. Seu nome significa American Standard Code for Information Interchange. Baseado no alfabeto inglês, originalmente usava apenas 128 símbolos (7 bits), não possuindo os caracteres acentuados de outras línguas.

[3]Unicode Transformation Format, que permite codificar 1.112.064 símbolos

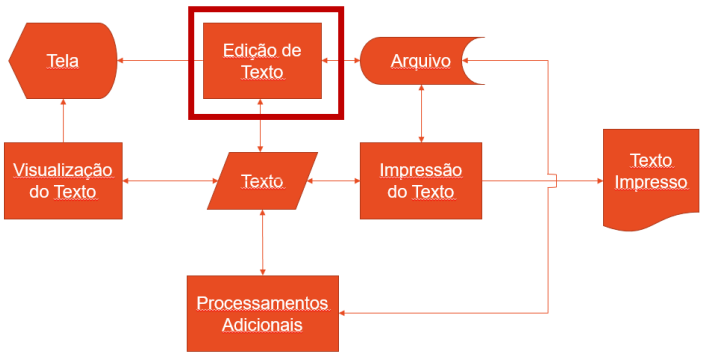

Figura 1.2: Função de um editor de texto na cadeia de processamento de texto.

por terminais ligados diretamente aos mesmos. Os primeiros editavam linha a linha, a seguir outros exigiam que o usuário gerenciasse o *buffer*, o seja, a parte do arquivo que estava em memória. Com o tempo chegamos a versões semelhantes as atuais.

Atualmente editores de texto tem um conjunto complexo de funções de busca, substituição, etc.

Exemplos de editores de texto atuais são o Notepad, que vem por *default* com o Windows, o `vi` e o `vim`, Notepad++, EditPlus, TextEdit e o poderosíssimo Emacs. A figura 1.3 mostra um exemplo to `vim`.

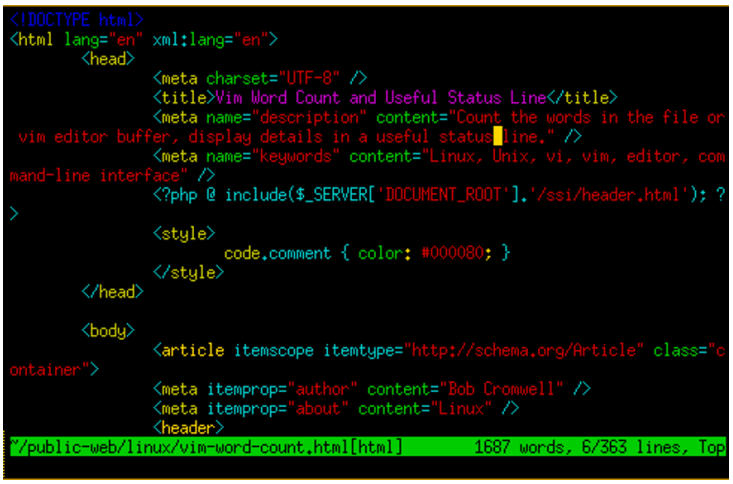

Figura 1.3: O editor de texto vim

1.1.2 IDEs

Uma IDE, ou "Integrated Development Environment", é uma extensão lógica da ideia de editor de texto criada originalmente para programadores, fornecendo serviços adicionais a edição de texto, como compilação, *debugging*, controle de versão, normalmente por meio de interfaces com os programas que fazem isso.

Uma IDE cobre a parte de edição e processamento de texto da cadeia de processamento de texto, como visto na figura 1.4.

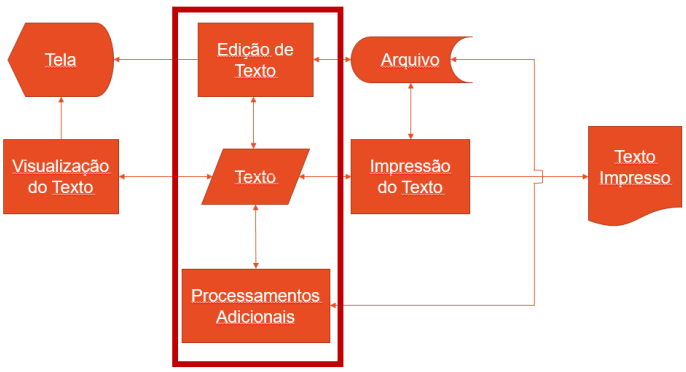

Figura 1.4: Um IDE permite editar e invocar outros processsamentos de texto

Exemplos de IDE são o Atom, o Visual Studio, o TEXStudio e o próprio Emacs. Com a evolução dos editores de texto, a fronteira entre editores e IDEs ficou indefinida, muitas vezes dependendo do uso que o usuário faz do programa. A figura 1.5 mostra um exemplo do TEXStudio.

1.1.3 Sistemas de Composição (*Typesetting*)

Já que era possível editar textos, porque não imprimi-los de forma adequada? Essa ideia levou a criação de programas de tipografia, que faziam a tradução de um arquivo texto, com marcações adequadas, para um outro arquivo que fosse interpretado em uma impressora (ou outras máquinas mais sofisticadas de *typesetting* dgital).

Um sistema de tipografia cobre apenas a parte de impressão da cadeia de processamento de texto, como visto na figura 1.6.

Essas marcações adequadas são como comandos, e um arquivo de texto marcado se assemelha a um programa de computador, por possuir palavras código que dão os comandos necessários. Sistemas desse tipo não possuem editores associados e são normalmente ativados por linha de comando.

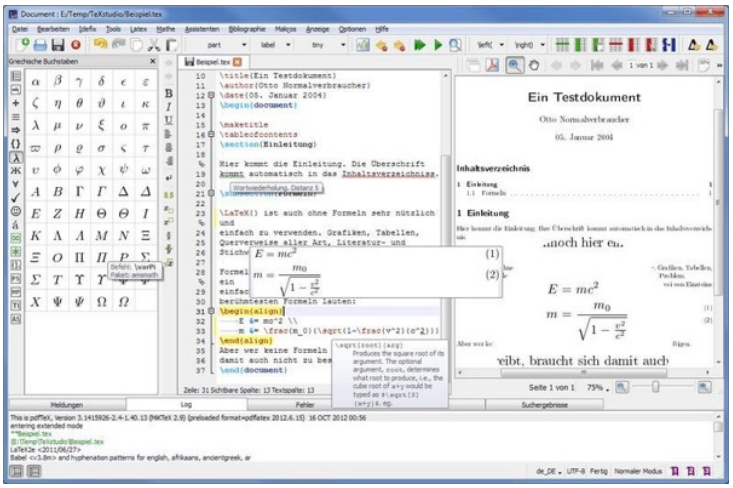

Figura 1.5: O TEX Studio

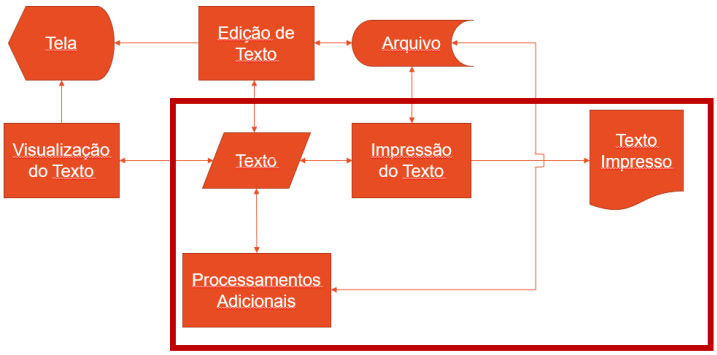

Figura 1.6: Sistemas de tipografia imprimem e fazem processamentos adicionais.

Exemplos de sistemas de tipografia são o troff, o TEX e o LATEX. A figura 1.7 mostra o LATEXsendo usado em linha de comando.

1.1.4 Processadores de Texto

Principalmente com o advento do micro-computador, ficou claro que uma das principais utilidades do computador seria permitir a criação de textos a serem publicados, logo um editor de texto deveria ser estendido para suportar funções como colocar palavras em negrito, itálico, selecionar fontes, etc.

Processadores de texto cumprem grande parte das funções na cadeia de processamento de texto, como mostra a figura 1.8.

Com a evolução dos terminais de computadores, micro-computadores e

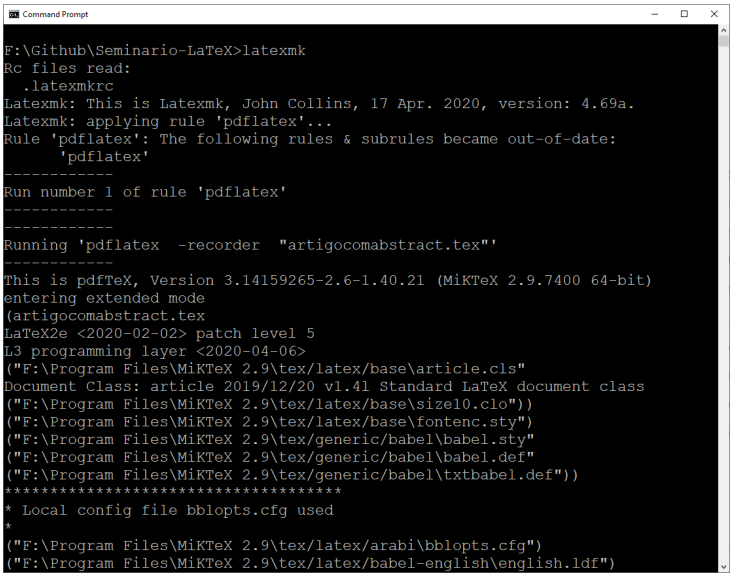

Figura 1.7: O LaTeXsendo usado

placas de vídeo e monitores, os processadores de texto evoluíram de sistemas que mostravam alguma coisa do que estava sendo prevista para o texto, como caracteres em bold, para sistemas que permitiam uma visualização prévia, até chegar a edição pelo conceito de WYSIWYG, lido "uiziwig", que mostra na tela quase que exatamente o que será visto na edição final, sendo as diferenças mínimas e quase imperceptíveis causadas por questões tecnológicas e da diferença entre papel e tintas e monitores.

Atualmente o Word é o processador de texto que domina amplamente o mercado, e seus principais concorrentes são sistemas de código aberto, como o LibreOffice Writer ou o Apache OpenOffice Writer. Um tela do Word é

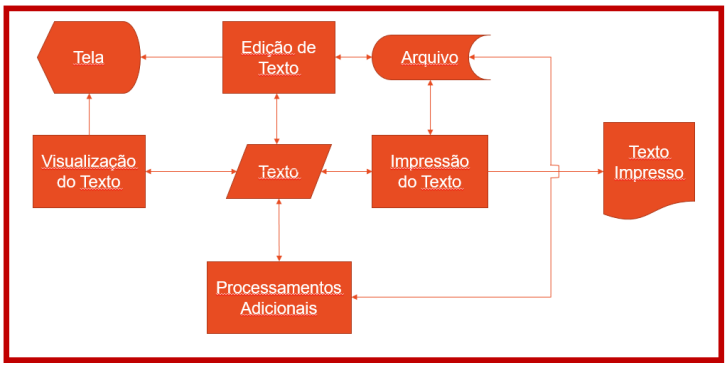

Figura 1.8: Os processadores de texto na cadeia de processamento de texto.

mostrada na figura 1.9.

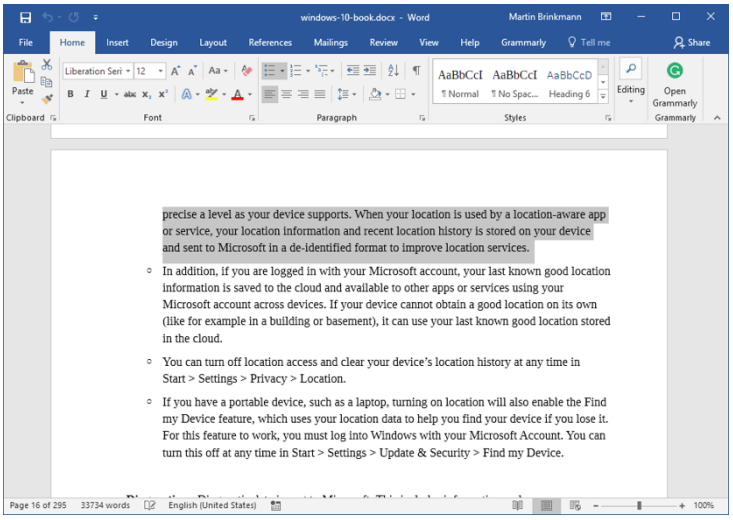

Figura 1.9: O Word é o principal processador de texto do mercado.

1.1.5 Sistemas de Autoria

Sistemas de autoria são uma evolução interessante dos processadores de texto, ou dos IDEs, voltadas para autores de livros, roteiros, etc. que não só permitem editar o texto, em formatos específicos, como também guardar informações como fichas de personagens, descrições de cena, *storyboards*, etc... Eles podem cobrir toda a cadeia de processamento de texto.

Exemplo de sistemas de autoria são o Scrivener, apresentado na figura 1.10, o Final Draft e o Celtx.

1.1.6 Sistemas de Publicação

Sistemas de publicação, ou*Desktop Publishing*, são sistemas cujo o foco é a composição de publicações, fornecendo uma capacidade de processamento de texto. São nesses sistemas que a maioria de jornais e revistas, e tamém manuais, folhetos, e outras formas de documentos impressos, são hoje preparados para a impressão.

Exemplos típicos de sistemas de publicação são o Publisher e o Framemaker, que é mostrado na figura1.11.

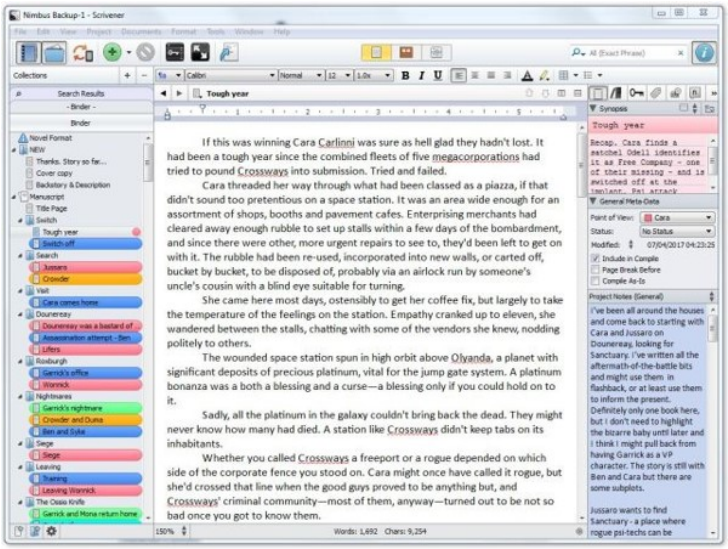

Figura 1.10: Uma tela do Scrivener

1.1.7 Sistemas Colaborativos de Edição

Esses sistemas aparecem cedo, porém se expandem principalmente com o fortalecimento da internet. Eles permitem que mais de uma pessoa edite um arquivo ao mesmo tempo. Atualmente o mais conhecido é o Google Docs, que é um processador de texto limitado em funcionalidade mas muito fácil de usar.

O ShareLateX foi um sistema colaborativo de edição voltado para o LaTeX que acabou sendo responsável por um renascimento do uso do LaTeX na academia. Acabou sendo incorporado ao concorrent Overleaf, que aparece na figura 1.12.

1.1.8 O que é o LaTeX

LaTeXé um sistema de typesetting baseado no TeXe apoiado com outros programas, como o biber que permite usar arquivos de texto marcados para criar arquivos a serem impressos, ou no formato PDF, seguindo regras de composição (*typesetting*) e usando fontes detalhadamente criadas para reproduzir a qualidade de fontes utilizada na composição manual, incluindo especialmente as ligaduras, que são caracteres especiais que representam dois ou mais caracteres de forma visualmente mais elegantes, como mostradas na figura 1.13.

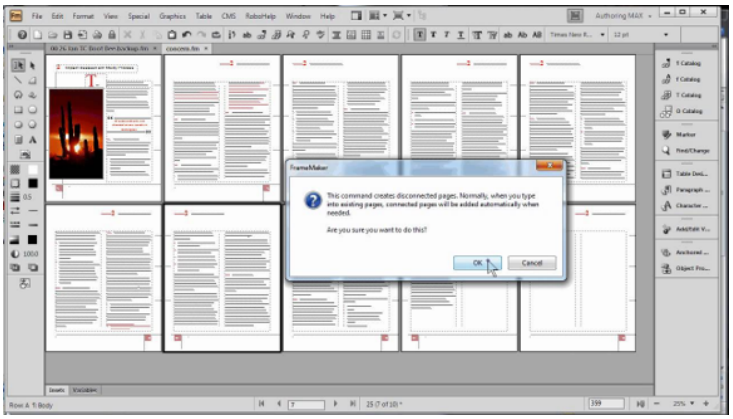

Figura 1.11: O software Framemaker

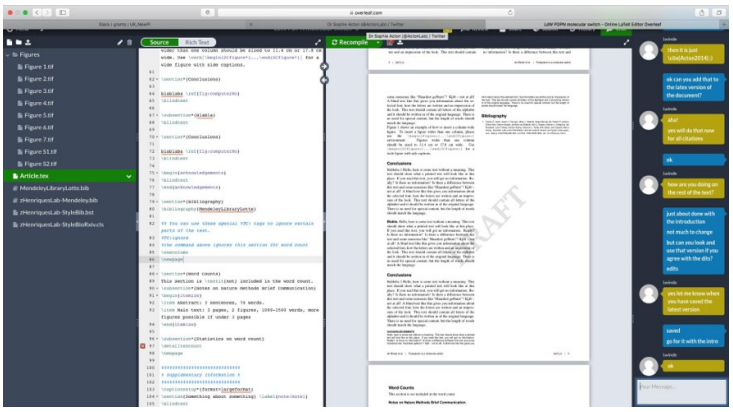

Figura 1.12: O Overleaf.

1.2 Tipos de Linguagens para Arquivos de Texto

Arquivos de texto podem ser guardados de várias formas. No limite, podemos fotografar uma página de texto e guardar a imagem, mas geralmente queremos um arquivo que possa ter seu texto manipulado.

Normalmente, na cadeia de processamento de texto, o arquivo é lido e colocado em memória em um formato mais adequado, e depois, quando salvo, é "rearrumado" de alguma forma, um formato de arquivo texto. Por exemplo, ao salvar um arquivo Word você pode escolher o seu formato nativo (.docx) ou vários outros formatos alternativos, como RTF[4], ou mesmo um arquivo texto, nesse caso perdendo toda a formatação.

[4]Rich Text Format

$$
\begin{array}{lcllcl}
fi & \longrightarrow & \textit{fi} & AE & \longrightarrow & Æ \\
ff & \longrightarrow & \textit{ff} & ae & \longrightarrow & æ \\
ffi & \longrightarrow & \textit{ffi} & OE & \longrightarrow & Œ \\
fl & \longrightarrow & \textit{fl} & oe & \longrightarrow & œ \\
ij & \longrightarrow & \ddot{y} & LATEX & \longrightarrow & \LaTeX
\end{array}
$$

Figura 1.13: Exemplos de ligaturas de fontes do TEX.

Os formatos de arquivo podem ser divididos em:

- Linguagens de Impressão/Visualização
  - PostScript, DVI, PDF
- Linguagens Intermediárias (de impressão)
  - .dvi
- Linguagens de Marcação
  - SGML, HTML, TEX LATEX Markdown, RPF, fods

O fato de uma linguagem de marcação ser legível por humanos não quer dizer que seja facilmente legível, principalmente quando geradas por máquinas. A figura 1.14 mostra um exemplo de arquivo Postscript.

```
%!PS-Adobe-3.0
%%Title: Datamatrix Barcode
%%Creator: Jp6raph Barcode http://www.aditus.nu/jpgraph/
%%CreationDate: Sun 5 Jul 23:06:27 2009
%%DocumentPaperSizes: A4
%%EndComments
%%BeginProlog
%%EndProlog

%%Page: 1 1

%Module width: 3 pt

%Data for bars. Only black bars are defined.
%The figures are for each row and in format: [xpos]
%Data: A Datamatrix barcode
3.05 setlinewidth
[[0] [6] [12] [18] [24] [30] [36] [42] [48] [54]] {{} forall 60 moveto 0 -3.05 rlineto stroke} forall
[[0] [6] [12] [18] [21] [33] [36] [39] [42] [51] [54] [57]] {{} forall 57 moveto 0 -3.05 rlineto stroke} forall
[[0] [18] [21] [24] [36] [42] [51]] {{} forall 54 moveto 0 -3.05 rlineto stroke} forall
[[0] [12] [15] [24] [30] [36] [39] [42] [45] [48] [51] [54] [57]] {{} forall 51 moveto 0 -3.05 rlineto stroke} forall
[[0] [6] [9] [21] [24] [33] [36] [51] [54]] {{} forall 48 moveto 0 -3.05 rlineto stroke} forall
[[0] [3] [6] [15] [18] [21] [24] [33] [48] [51] [57]] {{} forall 45 moveto 0 -3.05 rlineto stroke} forall
[[0] [9] [12] [15] [24] [27] [39] [51] [54]] {{} forall 42 moveto 0 -3.05 rlineto stroke} forall
[[0] [6] [18] [24] [27] [33] [39] [42] [48] [54] [57]] {{} forall 39 moveto 0 -3.05 rlineto stroke} forall
[[0] [9] [12] [15] [21] [24] [27] [30] [33] [39] [45] [54]] {{} forall 36 moveto 0 -3.05 rlineto stroke} forall
[[0] [3] [15] [18] [21] [24] [30] [33] [36] [39] [48] [57]] {{} forall 33 moveto 0 -3.05 rlineto stroke} forall
[[0] [3] [6] [18] [21] [27] [39] [45]] {{} forall 30 moveto 0 -3.05 rlineto stroke} forall
[[0] [15] [18] [21] [27] [30] [33] [36] [42] [51] [54] [57]] {{} forall 27 moveto 0 -3.05 rlineto stroke} forall
[[0] [3] [6] [12] [15] [18] [21] [30] [33] [36] [39] [42] [45] [48]] {{} forall 24 moveto 0 -3.05 rlineto stroke} forall
[[0] [6] [12] [27] [30] [42] [57]] {{} forall 21 moveto 0 -3.05 rlineto stroke} forall
[[0] [3] [9] [15] [18] [24] [33] [39] [42] [45] [48]] {{} forall 18 moveto 0 -3.05 rlineto stroke} forall
[[0] [9] [12] [21] [27] [30] [36] [39] [42] [45] [57]] {{} forall 15 moveto 0 -3.05 rlineto stroke} forall
[[0] [9] [12] [24] [39] [42] [48] [51] [54]] {{} forall 12 moveto 0 -3.05 rlineto stroke} forall
[[0] [9] [21] [27] [36] [39] [42] [45] [51] [57]] {{} forall 9 moveto 0 -3.05 rlineto stroke} forall
[[0] [15] [21] [24] [27] [36] [39] [42] [45] [54]] {{} forall 6 moveto 0 -3.05 rlineto stroke} forall
[[0] [3] [6] [9] [12] [15] [18] [21] [24] [27] [30] [33] [36] [39] [42] [45] [48] [51] [54] [57]] {{} forall 3 moveto 0 -3.05 rlineto stroke} forall

%End of Datamatrix Barcode

showpage

%%Trailer
```

Figura 1.14: Exemplo de um arquivo PostScript

1.2.1 Linguagens de Marcação

Um linguagem de marcação é caracterizada por um conjunto de códigos que é aplicado sobre o texto, ou sobre qualquer forma de dados, com o fim de adicionar informações específicas sobre esse texto, a cada trecho específico, como por exemplo a forma de exibi-lo graficamente.

Linguagens de marcação são uma forma simples de indicar como um texto deve ser impresso, e são usadas desde o início da digitalização do processo de composição e impressão.

As linguagens de marcação padronizadas permitem que programas diversos cumpram a mesma função. Elas podem ser divididas em:

- Procedurais
  - troff, TEX , LATEX, Postcript
- De apresentação
  - Wikis, Markdown
- Descritivas
  - SGML, HTML, XML

Linguagens de marcação procedurais normalmente incluem instruções de composição, ou seja, elas mantém unificadas a estrutura e a apresentação, enquanto linguagens de apresentação não se preocupam com nada além da apresentação imediata. Já linguagens descritivas, introduzias por Scribe, separam a estrutura da apresentação. LATEX, apesar de ainda conter instruções de composição, foi fortemente influenciada por essa ideia, e o usuário final praticamente só usa instruções que falam sobre a estrutura do documento. Um mapa mental das linguagens pode ser visto na figura 1.15(**Adams:2007**).

Linguagens de marcação normalmente são criadas para serem simultaneamente compreensíveis para o ser humano e tratáveis por um programa de computador.

Um exemplo de código LATEX e seu resultado é mostrado na figura 1.16.

1.2.2 Linguagens de Impressão

Linguagens de impressão tem como finalidade servir apenas para o processamento por um mecanismo de impressão ou um software específico de visualização. Nesse caso, não há preocupação com a comprensão do formato por seres humanos.

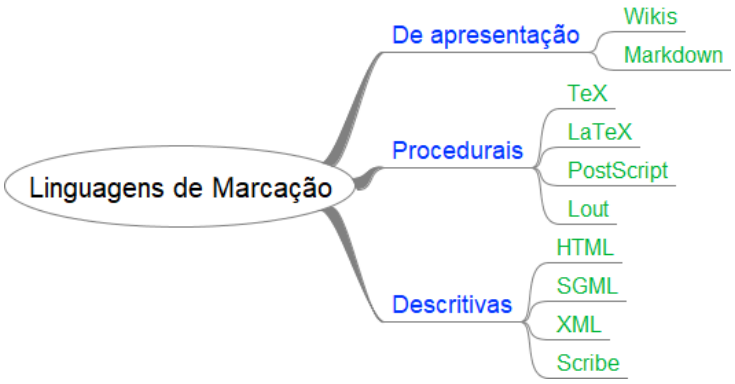

Figura 1.15: Mapa mental das linguagens de marcação.

```
\textbf{Um exemplo}
```

Um exemplo

Figura 1.16: Exemplo de código LaTeX

Exemplos de linguagens de impressão são o formato PDF, que é um Postscript empacotado de forma a descrever documentos compostos de páginas.

TEXtrouxe o conceito de uma linguagem de impressão intermediária, por meio do formato DVI, que torna o arquivo gerado pelo TEX independente do processamento final, o que permite que cada fabricante, ou interessado em geral, faça um programa próprio de conversão de DIV para qualquer formato de impressão. Logo, documentos DVI podem ser transformados em PDF, PS ou outro formato por software especiais, mas atualmente as implementações de LATEXtornam isso transparente.

1.3 Sistemas Mais Usados na Computação

Três sistemas são hoje muito usados na Computação:

- **Word**
  - Um software WYSIWYG
  - Líder do mercado
  - Superpoderoso
  - Difícil de usar para trabalho colaborativo

- **Google Docs**
  - Quase WYSIWYG
  - Sucesso entre os jovens
  - Pouca capacidade de plena expressão gráfica
  - Ótimo para trabalho colaborativo
- LaTeX
  - Melhor imagem de texto, mas no detalhe
  - Ótimo para Matemática
  - Difícil de usar
  - Empoderado pelo Overleaf
  - Renovado com os sistemas colaborativos

1.3.1 Mitos e Fatos

A Qualidade de saída do LaTeXé muito melhor

Verdade, mas só para quem entende o que está vendo. A qualidade da saída proporcionada pelo TeX depende de algum conhecimento do que é uma boa fonte, uma boa distribuição de texto. É possível que um leitor tenha alguma sensação de beleza ou conforto superior quando vê um texto gerado pelo TeX , mas é possível também que não saiba diferenciar, pois são detalhes.